ALPHA DISC LTD

FOR
DISC DRIVES

DISC DRIVE OWNERS HANBOOK

A step-by-step guide for use with the
BBC microcomputer.

LIST OF CONTENTS

NOTE:

Insert Diskette, Label upwards, push button in on front of drive, on 221 models LED comes on in half brilliance immediately power is applied.

Once drive is selected LED shows up in full brilliance.

LED GREEN = Tk.
LED RED = 80 Tk.

PREFACE

STEP-RATE OPTIONS

On the BBC Micro there are Link Options to increase the step rate of the disc drive.

When using the Canon MDD 220 or MDD 221 Drive, select the fast rate by inserting links 3 and 4 (4 ms). Only use the 4 ms step rate when the drive is in the 80 track Mode (i.e. L.E.D. RED).

When you wish to use the drives in the 40 Track Mode (i.e. L.E.D. GREEN) the drive must step at a slower speed. However the links do not have to be reset if you follow the following sequence:

Select Drive in 40 Track Mode (L.E.D. GREEN).

TYPE IN ★FX 255,255 (FOLLOWED BY BREAK KEY).

Then insert the 40 Track diskette and use as normal.

N.B. The drive will now operate in 80 Track Mode at the slower step rate.

This step rate will be cancelled when the BBC Micro is switched off.

INTRODUCTION

The minimum requirements for a disc drive system are as follows:

A model B microcomputer with the disc interface.
A disc drive with ribbon and power cables.
A utility disc.

The utility disc allows you to format your own blank diskettes, which must be done prior to use.

This manual provides a guide to the basic concepts behind using disc drives with the BBC microcomputer and references for the BBC commands and error messages which are used to control the disc unit. It explains the use of standard and special files and the way files are laid out on the disc. It also gives instructions on the use of the ALPHA DISC utilities program.

You should be familiar with the BBC computer and the BBC User Guide.

DISC SYSTEM BASICS

What are disc drives?

Disc drives are quite simply another form of memory that the computer can use to store your programs and data. This section will describe the disc drive and compare it with the other methods of storage available with the computer.

When you turn your BBC on and type in a program, the computer automatically stores it in RAM. RAM stands for random access memory and this type of memory has the advantage that the computer can access any part of it very quickly. It has the disadvantage that in order to store data it requires the power to be on. When the computer is switched off, all the data disappears. This type of memory is called 'volatile'. The BBC has the capability to transfer the data in the RAM to 'non-volatile' forms of storage such as cassette tape or disc.

The BBC User Guide explains the use of a cassette recorder to save programs on tape. The idea is the same for disc drives, but the disadvantages of using cassettes are overcome.

What are these disadvantages?

1. The tape is slow not only in saving and loading programs, but also in winding the tape to the correct position before the program can be saved or loaded.

2. You must note where each program is saved on the tape so you do not accidentally overwrite it with another program.

3. You must ensure that the tape is wound to the correct position.

4. All data is stored sequentially on the tape, (i.e. data 1, data 2, data 3 etc.) and must be read in the same way, even if you only want to read data 3. This is a very serious disadvantage when you start to write more advanced programs.

The disc system will allow data or programs to be transferred at as much as one hundred times the speed of the tape system. In addition, the computer will automatically keep a record of the locations of your programs on the disc. All you need to do is to type 'load' or 'save' and the computer does the rest. This means, when writing or saving programs you are able to access data almost instantaneously.

In order to explain how the disc drive works it is first necessary to examine its basic functions.

FIGURE1

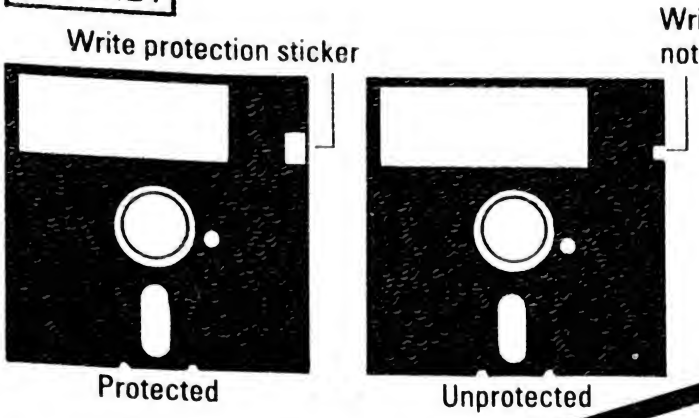
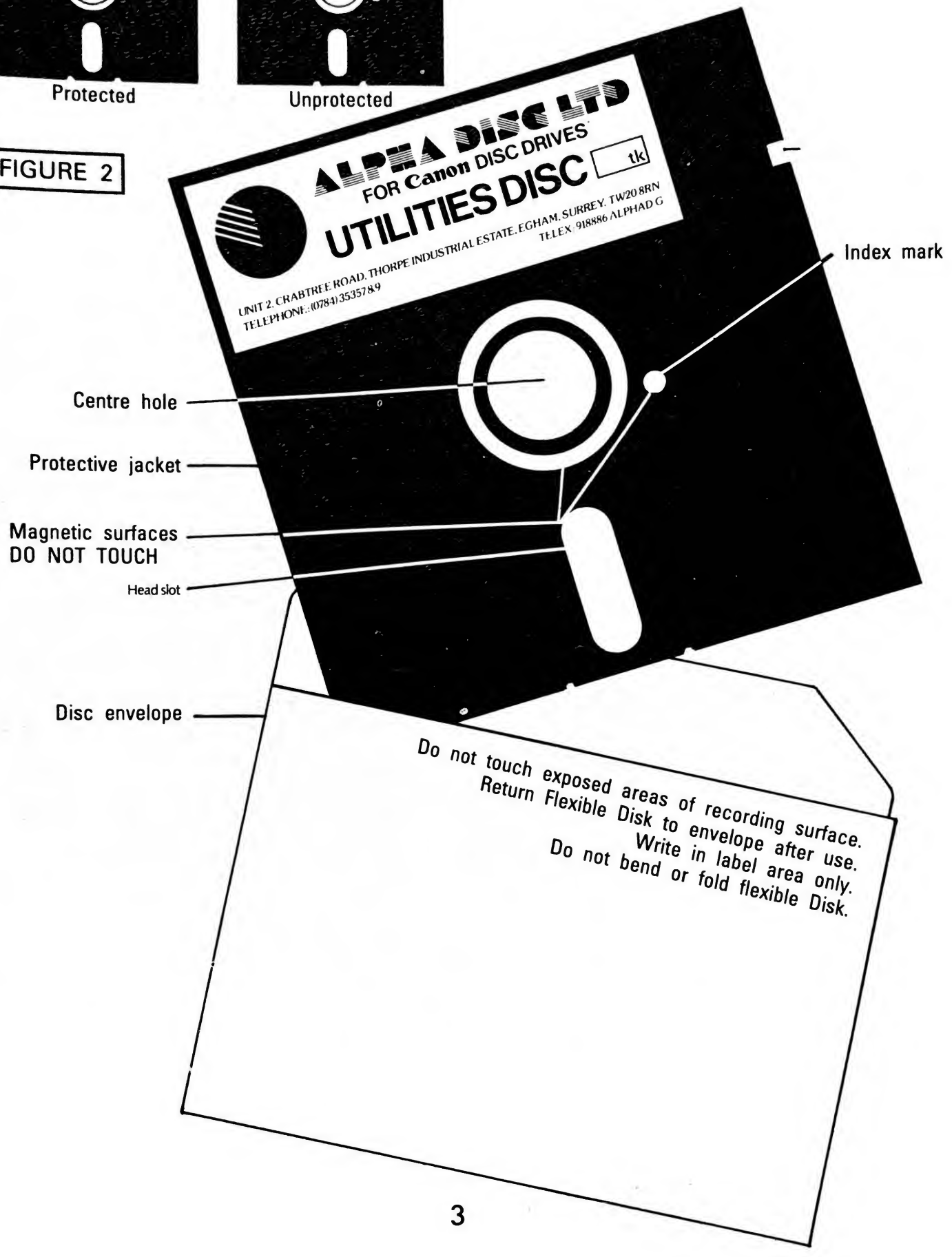


FIGURE 2



THE FLOPPY DISC AND THE DISC DRIVE

The type of floppy disc used on the BBC computers is illustrated in figure 1. The floppy disc itself is made from a type of plastic called mylar and has an oxide layer bonded to each side. The floppy disc is sealed within a square protective jacket, and when not in use, should be stored in its dust cover as shown in figure 2.

The protective jacket has holes and slots cut into it. The central hole is circular, allowing a rotating boss from the disc drive motor to spin the floppy disc within its protective jacket. There are two long slots on each side to allow the magnetic read/write head to rest on the surface of the disc when a read or write operation is being performed. There is a small circular hole in the floppy disc jacket and a corresponding hole in the floppy disc. When these two holes are in line, an infra red light passes through them. The detection of this light causes a pulse known as the index. The index pulse is used as a reference point to locate data.

The last notch in the side of the jacket is the write-protect notch. It works in the same way as the slot in the back of cassette tapes or video tapes to prevent over recording. Write protection is achieved by covering this notch with a write protect tab.

Handling discs

This is important. Floppy discs are a very reliable method of storing information provided they are treated properly.

Here is how you should treat your discs:

Keep them in their dust covers when not in use.

Store them in a box, in a dry, cool place.

Write on the labels only with a felt tip pen, and press very gently.

Do not touch the surface of the floppy disc, hold it only by its protective jacket.

Never fold or bend the floppy disc.

Do not leave the disc in a magnetic field (i.e. on top of televisions, radios, telephones, tape recorders, etc.).

Make sure when purchasing floppy discs that they are compatible with your disc drive(s), i.e. 40 track / 80 track, single-sided / double-sided.

The drive unit

Disc drives supplied by ALPHA DISC are of the following basic types:

Single-sided, 40 track.

Double-sided, 40 track.

Doubled-sided, 40/80 track switchable.

All disc drives supplied by ALPHA DISC are capable of operating in single or double density without any modifications.

ALPHA DISC supply single disc drive sub-systems both with or without internal power supply. All dual disc drive sub-systems are supplied with an internal power supply. We follow this policy as under certain circumstances two drives run from the BBC Micro power supply can cause data transfer errors.

How does the disc unit work?

The best comparison is with a filing cabinet. After all, the disc unit is just an automatic filing cabinet. The drive itself is like an empty cabinet. When a disc is inserted into the drive, then the cabinet has a set of drawers. If another disc is put in, the cabinet has an entirely different set of drawers.

When you use a cabinet to store information, it is sensible to have a well-organised and standard way of storing and labelling the information. In addition there should be an index to tell you where information is stored. This makes it easier for you to find the information you want, and allows other people to use the cabinet as well. So you put dividers into each drawer and label them, and one drawer you allocate as the index to the rest of the cabinet. The same idea is used in organising discs before they can be used. The process is called 'formatting'.

Disc format splits the disc up into 'tracks' and 'sectors'. These could be compared to 'drawers' and 'dividers'. Each sector or divider can only hold so much information before it spills over into the next. Each track or drawer can only hold so much before spilling into the next. When the whole disc is full, you must either throw some information away, or use a new disc.

On the BBC compatible format, each sector can store 256 bytes, and there are 10 sectors to every track. The disc may have 40 or 80 tracks depending on the type of drive. This gives you 102,400 or 204,800 bytes.

Programs or data saved to the disc are stored on sequential sectors, starting with the first empty sector. If the last sector used is only partially full, then the rest of that sector will hold 'dummy' data, so that data is always saved to a whole number of sectors. That unused portion of the last sector is not used for storing anything else.

To read or write a sector, the drive starts the disc spinning, moves its read/write head to the correct track, and waits until the appropriate sector arrives under the head. Then as it passes, the head will either read or write data by detecting or altering the pattern of magnetic 'bits' on the oxide surface of the disc.

How does the drive know what track and sector it's on?

The formatting process we mentioned earlier labels each sector with its sector and track number. When the drive is selected the read/write head always recalibrates back to track Ø (the outermost track). The head then steps to the appropriate tracks as directed by the computer's DFS, (Disk Filing System).

What is the DFS?

This stands for the Disc Filing System, which is a machine code program stored in ROM (Read Only Memory) in the BBC. ROM is a permanent or 'non-volatile' memory, so the DFS is always resident in the computer, even when the computer is switched off.

The DFS is responsible for the procedures for reading and writing data on the floppy disc.

The head starts at track Ø. It looks at the first two sectors (Ø and 1). This area is known as the catalogue, where the locations of all the stored data files are kept.

(a) Writing data from the catalogue the DFS locates suitable free space for the program to be stored. The head then steps to the appropriate track(s) and sector(s) and writes the data. Also a record is made in the catalogue as to the location of that program.

(b) Reading data: From the catalogue the DFS locates the program. The head then steps to the appropriate track(s) and sector(s) and reads the data.

FIG 3 Units with mains PSU

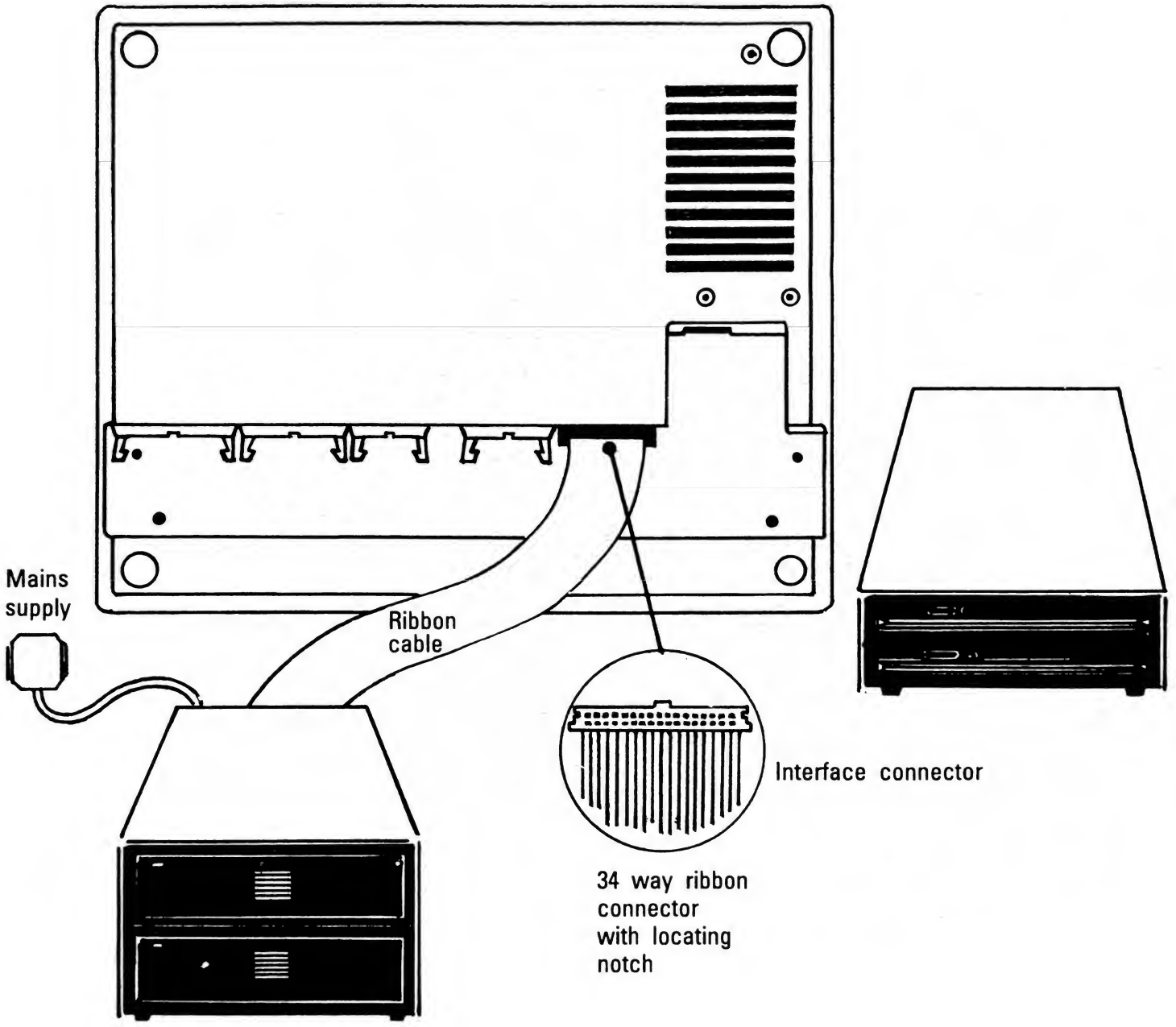
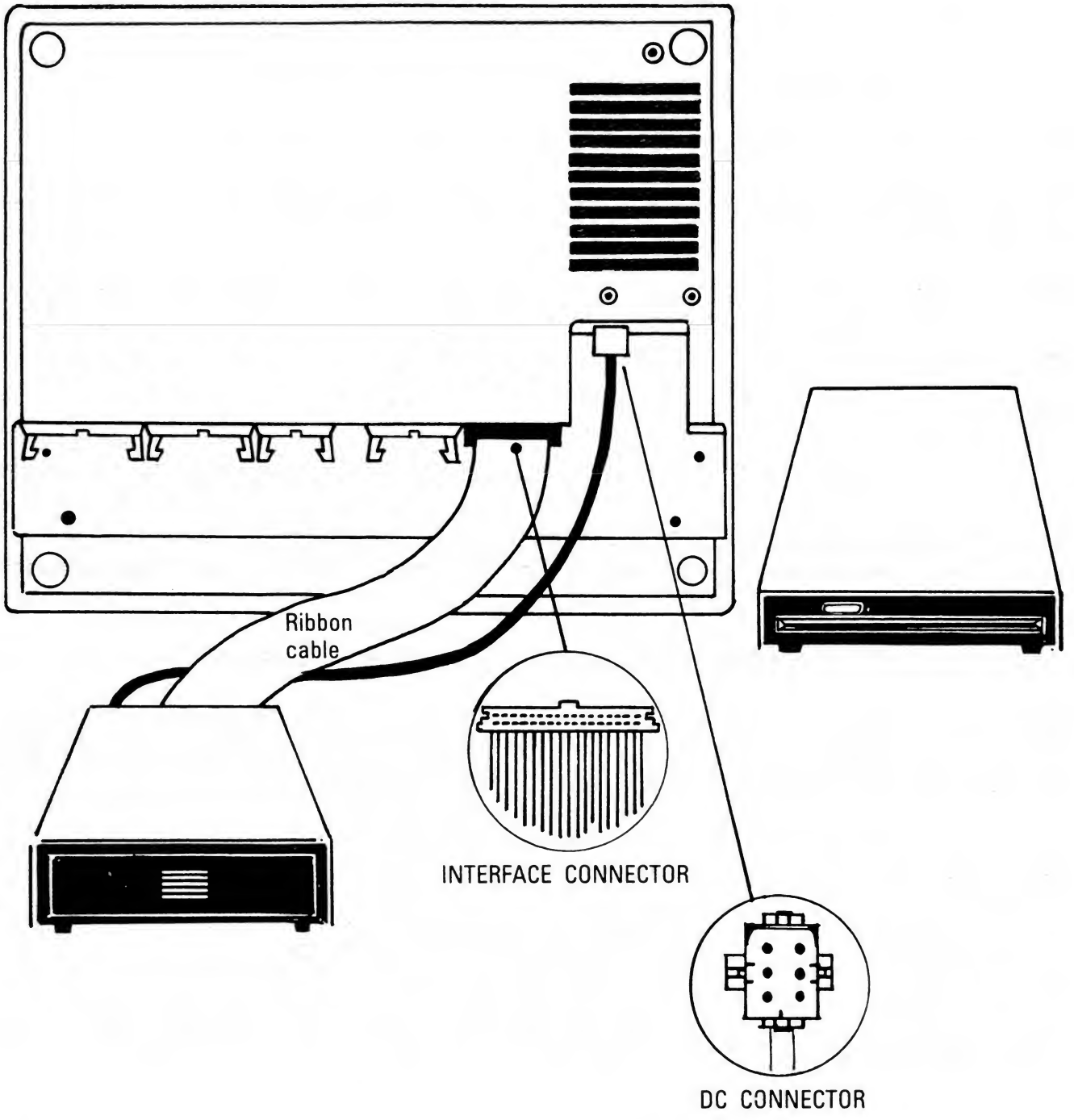


FIGURE 4 units using BBC PSU



NB All drives using BBC PSU have a white DC connector to the power cable. **DO NOT CONNECT THIS LEAD TO THE MAINS SUPPLY**

STARTING

Connecting the drive unit. (Please read the whole section before connecting your disc drive).

Make sure that the computer is switched off.

Disc drive systems with internal power supply: Follow figure 4.

Disc drive systems without internal power supply: Follow figure 3.

Power up the computer and drive unit and press (BREAK). The following message should appear.

BBC Computer 32K

Acorn DFS

BASIC

>

If it does not, or the red activity lights stay on, the most likely fault is that the ribbon cable is connected the wrong way round. Power off, reverse the cable and try again.

If you have this message, it indicates that the DFS is working.

Before going on to the next part, it is very important that you write-protect the ALPHA DISC UTILITIES disc by sticking the tab over the write-protect notch. This will prevent accidentally overwriting the utilities programs.

Now insert your ALPHA UTILITIES DISC into the disc drive (lower disc drive in a dual system). This is drive Ø.

Now press (SHIFT) and (BREAK) together and then release the (BREAK) key before the (SHIFT) key. The drive motor will turn on, the head will load and the activity light on the front of the drive will come on. The utilities disc will then load and the ALPHA DISC logo will appear on the screen. If this does not happen, check all the cable connections.

Copying the utility disc

We strongly advise that you keep at least one copy of the utilities disc as this is a valuable program.

Now if you have turned the machine off, follow the steps given above until you are at the point where you have the ALPHA DISC logo on the screen. Remove the utilities disc and insert a suitable blank floppy disc into one of the disc drives.

Your drives will be 40 or 80 track. From the menu prompt, select the appropriate option for your drive:

A. FORM 40
B. FORM 80
C. VERIFY

The screen will clear and the following will be displayed:

e.g. ALPHA DISC LTD.
FORM 40

ENTER DRIVE NUMBER (0–3):
READY TO FORMAT Y/N
PLEASE WAIT . . .

(Formatting procedure now starts and indicates the tracks being formatted e.g. 0 1 2 3 4 etc.)

As each track is formatted it is also verified to check that the format is correct.

If a ? appears after any displayed number, this means that there has been between 1 and 10 retires of writing to this track. Try the formatting procedure again. If the problem persists on the same track(s), try an alternative disc.

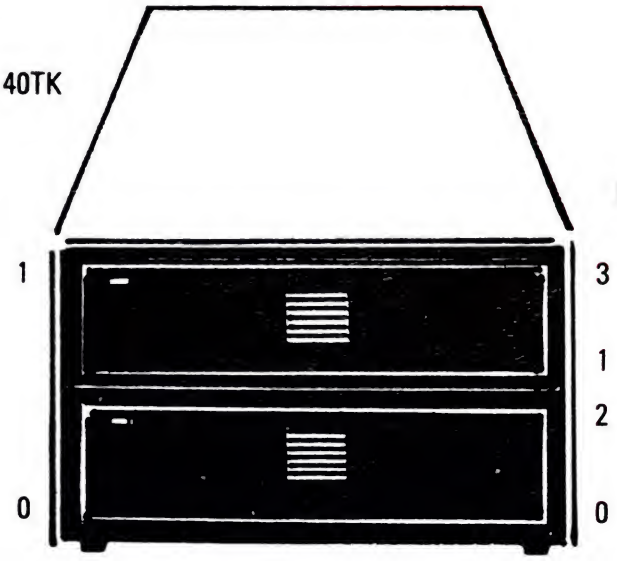
If the formatting routine will not start, or stops after any number before the last track number (39 or 79 as appropriate), this means there is a possible serious fault.

Once the format program is finished, you are ready to copy your utilities disc. Decide which drives you wish to copy from (source) and to (destination), e.g. from drive 0 to drive 1.
For an explanation of disc drive numbering on the BBC Micro, see figure 5.

FIGURE 5

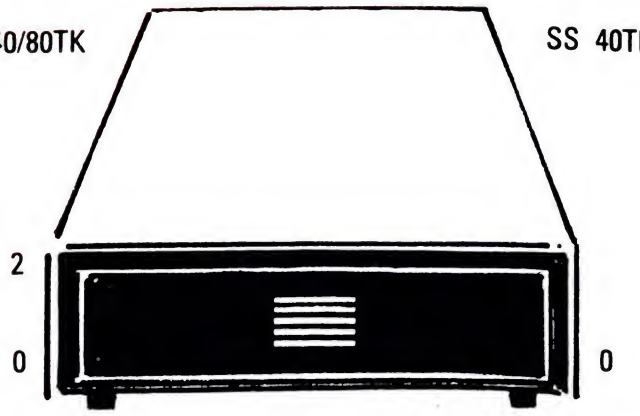
DRIVE CONFIGURATION ON BBC

SS 40TK

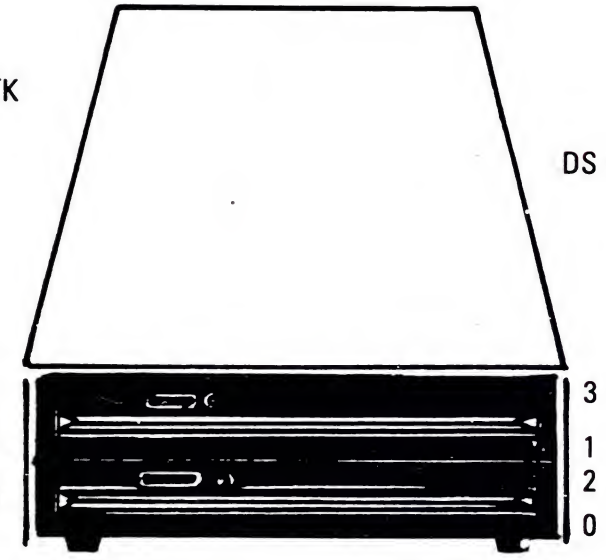


DS 40/80TK

SS 40TK

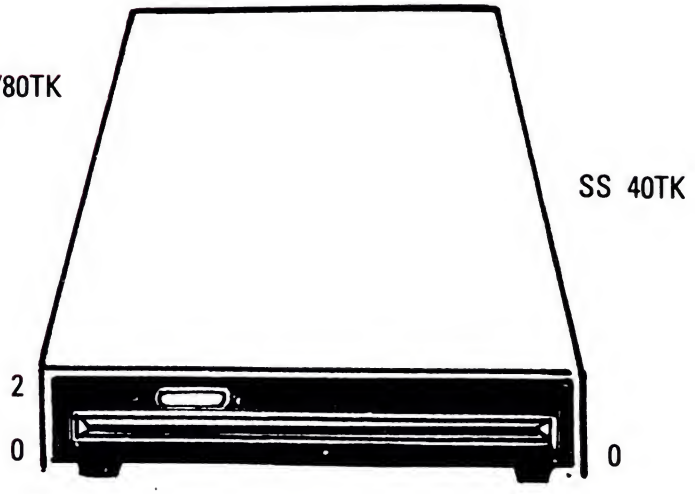


DS 40TK



DS 40/80TK

SS 40TK



Take care when placing DISC in drive.
Have label facing up. →

Type in:

★ ENABLE
★ BACKUP 0 1

The computer will make a copy of the utilities disc. When it is finished it will return to the > prompt. Then remove the source utilities disc, return it to its dust cover and put it somewhere safe. In future use the copy you have made as your utilities disc.

It is possible to copy a disc using a single drive.

Type in:

★ ENABLE
★ BACKUP 0 0

(Follow the instructions displayed on the screen).

DISC FILES

The catalogue

Insert your utilities disc (the copy not the master) into drive Ø and press (BREAK). When the computer gives you the > prompt.

Type in:

★ CAT

You should get the following display:

```
(Name of disc)
(ØØ) (displays the number of times the disc has been written to)
        Drive :Ø                          Option :0 (or 1, 2, 3)
        Directory :Ø.$                    Library :Ø $


                !BOOT    L          A2        L
                ALPHA    L          LOGON     L
                M2       L
>
```

This is a summary of the catalogue that the DFS uses to keep track of all the programs stored on that disc.

To begin at the top: the disc can be named. This is more for your benefit than the computers, and help you in keeping track of which disc is a master and which is the copy for instance.

The drive indicated is number Ø, the directory and library as Ø.$. This is the default case when you power up. You can set and reset the directory as you program, and this has the useful feature that you are locked out of files which are not in the directory you have just set.

The directory name may be any single character. This is a security feature to prevent accidental overwriting of programs, and is in addition to the 'LOCK' feature which is shown applied to all the files on this utility disc.

The library feature allows access to programs in another directory, without explicitly changing the current directory. You can therefore access two directories on the same drive, or two directories on two different drives.

One more feature is shown in the directory. This is the 'OPTION' selection which may be Ø to 3. These options control the action of the computer when it is powered up. Further explanation is given in the section on DFS commands.

Now for some details on the specifications of files:

When you type:

SAVE "filename"

The DFS will save a file called filename on the drive which is currently in use and in the directory which is currently open. If you want to store the file somewhere else, you can specify the drive and the directory of your choice in the following manner:

e.g.

SAVE ":2.C.PROG"

This command will save a program called PROG on drive 2 and in directory C.

The directory name may be any single character. Default directory at power on is $. Default drive is 0.

The file name may be up to 7 characters long. However the characters # and ★ have special meanings as described below. Furthermore, the characters . and : are not allowed in filenames to prevent confusion between drive, directory and file name.

(Wildcard characters # and ★):

These are similar to wildcards in card games where one special card may take the value of any other in the pack. These are used to identify a group of file names as opposed to one unique name.

The usual way to refer to unique filenames is <fsp>, which stands for file specification. When a wildcard character is included, the file name becomes ambiguous and is referred to as the <afsp> which stands for ambiguous or alternative file specification. To give an example:

:0.$.MYPROG1
:0.$.MYPROG2
:0.$.MYPROG3

are all <fsp>.

The name
:0.$.MYPROG #

is an <afsp> and refers to all three programs.

How is this useful?

Some commands in the DFS allow an <afsp>or a <fsp> as the file specifications. For instance

★ INFO :0.$.MYPROG1

will produce some information relating to that program only.

★ INFO :0.$.MYPROG #

will produce information on all three files.

Similarly ★ is a wildcard character(s) of undefined length that is determined only by the preceeding characters. e.g. in the example above:

★ INFO :0.$.M ★

will produce information on all the three files

:0.$.MYPROG1
:0.$.MYPROG2
:0.$.MYPROG3

and in addition any other programs with the first letter M e.g. M2 or MINE or MONST

DISC FILING SYSTEM COMMANDS

The next section deals with all the commands that you can use with the DFS.

DFS commands are preceeded by the special character ★, to set them apart from the ordinary BASIC commands. When creating the copy of the utilities disc, you used two DFS commands ★ ENABLE and ★ BACKUP. You may have already experimented with the BASIC commands SAVE and LOAD. Note that these are different from the DFS commands ★ SAVE and ★ LOAD.

In describing the commands, various abbreviations are used. You have already met <fsp> and <afsp> in the previous section. The two others in this section are

<drv> which stands for a drive number i.e. 0, 1, 2 or 3

<dir> which stands for the directory name i.e. $, A-Z etc.

In addition to describing the syntax of the commands, and the effect that they have, this section will also give details of the abbreviated form of the commands. For instance.

★ DR.2 may be used instead of

★ DRIVE 2 to set the current drive to drive number 2.

Where appropriate, associated commands will be listed. For instance

★ BACKUP must be preceeded by

★ ENABLE

The command parameters will be given in angular brackets e.g. <drv> and also with round brackets where they are optional.

DFS COMMANDS IN ALPHABETICAL ORDER

★ ACCESS <afsp> (L)

This is a security command. It produces a 'lock' on a file which prevents it being deleted, overwritten or added to until you use the command again to 'unlock' the file. The only command which can defeat the lock is ★ BACKUP. The format program will also destroy the file.

Note that you can load a locked file, but most then save it under a different name, or unlock the disc file before saving the new version. This is because SAVE MYPROG overwrites any file called MYPROG on the disc.

Examples:

★ ACCESS MYPROG L	This will lock MYPROG
★ ACCESS MYPROG	This will unlock MYPROG

Abbreviation:

★ A. MYPROG L

If you attempt to delete or rename a locked file using any of the DFS commands except ★ BACKUP, you will get this message

File locked

If you use ★ ACCESS on a write-protected disc, you will get the message

Disc write-protected

★ BACKUP <source drv> <destination drv>

This reads all the data on one disc and makes an exact copy of it on another disc. It must be preceeded by an enable command before it will work.

EXAMPLE:

★ ENABLE
★ BACKUP 0 1

If you do not use the ★ ENABLE command first, the message

Not enabled will be displayed.

If you give 0 as the source and destination drives, the backup program will prompt you to alternately insert the source and destination discs. It will take 5 of those swops to copy an entire 40 track disc, and 10 for an 80 track.

The destination disc must of course be formatted before it can be used with ★ BACKUP or any other copying command.

★ BUILD <fsp>

This allows you to create disc files directly from the keyboard. After giving this command, everything you type in will go into the file you have named until you press the escape key. The build program will prompt you with line numbers.

Example:

```
★ BUILD TEST
      1 HELLO
      2 THIS IS A TEST
      3 BYE
      4 <esc>
```

will create a file called TEST on the disc. If you enter

★ TYPE TEST the machine will display the file contents.

```
HELLO
THIS IS A TEST
BYE
```

Abbreviation:

★ BU. TEST

This command is useful for creating EXEC files (see ★ EXEX) and the !BOOT file (see ★ OP4).

Further useful example:

Type in:

```
★ BUILD !BOOT
      1 CHAIN "MENU"
      2 [ESC]

★ OPT 4 3
```

On pressing [SHIFT] [BREAK] combination this will execute the file !BOOT and chain the selected program. In this example the MENU file.

★ CAT <drv>

The catalogue of all files on the drive is displayed.

Example:

```
★ CAT Ø
UTILCPY (1)
Drive:Ø                 Option:3 (Exec)
Directory :Ø.$          Library:Ø.$

          !BOOT    L          A2         L
          ALPHA    L          LOGON      L
          M2       L
A.TEST          B.MYPROG
```

The first line shows the name of the disc (UTILCPY) and the number of times the disc has been written to. Next line shows the drive which is currently accessible, and the power up option. In this case the power up or auto start option will 'exec' the file called !BOOT (see ★ EXEC and ★ OP4). Next line shows the directory and the library which are currently selected. These are the power on default selections for both (see ★ DRIVE and ★ LIB).

The block below the heading lists the files in alphabetical order. L after the filename indicates that they are locked (see ★ ACCESS). For further details on information about files on disc see ★ INFO.

The last line is a list of files which are on the disc but in different directories, the two examples being in directories A and B.

★ . (will display the catalogue of drive Ø)
★ .1 (will display the catalogue of drive 1)

★ COMPACT <drv>

Compacting a disc may produce space for more files. The reason for this is that in the course of saving and deleting files, the DFS may leave gaps between files. If all the files are shuffled so that all the gaps create one space at the end, this space may be quite considerable.

Example:

★ COMPACT1

As the DFS compacts the disc, it wil display file data as it shuffles each file.

Abbreviation:

★ COM. <drv>

★ COPY <source drv> <destination drv> <afsp>

This is another backup routine, with the important differences that it will only copy the files specified and will not overwrite any files that exist on the destination drive.

Example:

★ COPY 0 1 MYPROG1

will copy MYPROG1 from drive 0 to drive 1.

★ COPY 0 1 MY★

will copy all files that start with MY.

Abbreviation:

★ COP. <source drv> <destination drv> <afsp>

★ DELETE <fsp>

This will delete the named file from the current disc and directory. Locked files not in the current directory, files not on the current drive or files on write-protected discs will produce the appropriate refusual message, once deleted the information cannot be retrieved.

Example:

★ DELETE MYPROG1

Deletes MYPROG1 on the current disc and directory if it is not locked (see ★ ACCESS) or otherwise protected.

Abbreviation:

★ DEL. MYPROG1

★ DESTROY <afsp>

This is a version of delete for groups of files. The program will prompt you before destroying the group of files. The command must be preceeded by ★ ENABLE.

Example:

★ ENABLE
★ DESTROY :0.$.MYPROG #
$.MYPROG1
$.MYPROG2
$.MYPROG3

Delete (Y/N)?

When you type Y in response to this, all those files listed after the command will be destroyed. The program will display

Deleted

(when the task is completed).

Abbreviation:

★ DES. :0.$.MYPROG #

will delete the same files as above.

★ DIR (directory character)

Sets the currently accessed directory to whichever the directory character specifies. The directory can be set to any character.

Example:

★ DIR A

Sets to directory A. All saved programs will go to this directory and all loaded programs will come from this directory, unless another directory is explicity specified.

Abbreviation:

★ DIR <A>

Files not in the current directory may be accessed by preceeding the file name with the directory letter.

★ DRIVE <drv>

This sets the default drive to the number drv. Drv may take the values 0, 1, 2, or 3, assuming you have all those drives present in the system.

Example:

★ DRIVE 1

Sets the drive to 1. All following saves and loads will use this drive until another is specified. However, note that commands which explicitly refer to another drive will effect that drive. They will not reset the default drive number.

Abbreviation:

★ DR.1

★ DUMP <fsp>

This will list a file in its hexadecimal form on the screen.

Example:

★ DUMP MYPROG1

Abbreviation:

★ DU. MYPROG1

It is generally useful to set page mode for listing before using this command. <cntrl> sets page mode, <cntrl> 0 resets it.

★ ENABLE

This command is only used to preceed certain commands that are very powerful (see ★ BACKUP and ★ DESTROY).

Example:

★ ENABLE
★ DESTROY <afsp>

Abbreviation:

★ EN.

No other command must come between the enable and the destroy or backup command. Each time a destroy or backup is used, it must be enabled.

★ EXEC <fsp>

This command is very useful in that it allows you to use a disc file to control the computer. It works as if the contents of the file had been typed in at the keyboard. This means that if you have any sequence of commands that you use repeatedly, then you can make a file of them (by using ★ BUILD) and ★ EXEC that file every time you want to execute that sequence of commands.

Example:

★ EXEC SEQNC

will take the contents of the file SEQNC and submit them to the computer as if you had typed them in at the keyboard.

Abbreviation:

★ E. SEQNC

★ HELP <keyword>
This displays lists of the commands available in the DFS. The two keywords are DFS and UTILS.

Example:
★ HELP DFS (lists all the DFS commands that are available)

★ HELP UTILS (lists utilities available within the DFS)

The list does not include ★ EXEC, ★ LOAD, ★ RUN, ★ SAVE and ★ SPOOL. These commands actually operate outside of the DFS.

★ INFO <afsp>
This command will display special information about the group of files which correspond to the <afsp>. The information given is, in order of appearance across the screen:

The directory.
The filename.
The access restriction (if any).
The load address of the first byte of the program when it is loaded.
The execution address of the start of the program.
The length of the program in bytes.
The address of the first sector on which the program starts.

Example:
★ INFO MYPROG #

will produce a listing as described above of all files starting with MYPROG.

Abbreviation:

★ I. MYPROG #

★ LIB (:<drv>.) <dir>

This sets the library to the specified drive and directory. Once set, you have the option of entering just ★ MYPROG1 and the DFS will search the directory you specified on the drive you specified and run any program of that name.

Example:
★ LIB :1.A

sets library to drive 1 and directory A. Then the command

★ MYPROG has the same effect as

★ RUN :1.A.MYPROG

★ LIST <fsp>

This command displays the file named in the command as a text file on the screen, with each line numbered. Note that the commands in a BASIC program are stored in token fashion.

★ LIST of a BASIC program will reproduce the token form and will be most difficult to decode. You can however use ★ SPOOL to produce a text form of the program.

Example:

```
NEW
10 PRINT "HELLO"
SAVE "MYPROG"
★ LIST MYPROG
1
2    "HELLO"
3
```

It is useful for long listings to set page mode by typing <cntrl>N. Press shift for each page. Page mode will be reset by <cntrl>0.

★ LOAD <fsp> (<start address>)

This will read the named file from the disc and start it running from the address specified in the command, or at the load address of the file when it was saved.

Example:

★ LOAD MYPROG

loads and runs from the address defined when MYPROG was saved.

★ LOAD MYPROG 23FF

loads to 23FF (hex) and runs from there.

Abbreviation:

★ L. MYPROG 23FF

★ OPT 1 (n)

This command will set a marker in the computer so that every time it acesses a disc file, information about that file will be displayed. The information will be the same as described for the ★ INFO command.

Example:

★ OPT 1 1 or ★ OPT 1,1

will enable these messages.
★ OPT 1 Ø or ★ OPT 1,Ø

will disable them.

Abbreviation:

★ 0. 1 Ø

The space or comma between the two numbers are essential in this command.

★ OPT 4 (n)

This option allows for different actions which can be taken by the computer automatically when it restarts (after a SHIFT and BREAK). The options available are 0, 1, 2 or 3. Depending on which is set, the computer will load, run or exec a special file which must be called !BOOT, and must be in the directory $ of drive 0.

Example:

★ OPT 4 0	will not do anything
★ OPT 4 1	will load the file !BOOT
★ OPT 4 2	will run the file !BOOT
★ OPT 4 3	will execute the file !BOOT

Abbreviation:

★ 0.4 (n)

A space or comma between the 4 and (n) of the command is essential.

★ RENAME <old fsp> <new fsp>

This changes the name and directory of the specified file.
★ RENAME will not work with protected files.

Example:

★ RENAME OLDPROG NEWPROG

will rename a program called OLDPROG. The new name will be NEWPROG.

★ RENAME $.OLDPROG A.NEWPROG

will also change the directory under which the file may be accessed.

Abbreviation:

★ RE. OLDPROG NEWPROG

★ RUN <fsp> (parameters to utility)

This command is used to load and run machine code programs.

Example:

★ RUN MYPROG

will load the machine code program MYPROG and jump to the beginning of that program.

★ RUN MYPROG PARAM

will load and run the program, using the parameters to decide on a course of action:

Abbreviation:

★ MYPROG

will perform the same operation as the first example.

★ SAVE <fsp> <start address> <finish address> (<execute address>) (<reload address>)

This is not the same as the basic command 'SAVE'. ★ SAVE will store an area of memory and requires at least the start and finish addresses, or the length.

Example:

★ SAVE MYPROG 12FF 13FF 1300 12FF

will load the area of memory starting at 12FF (hex) and finishing at 13FF onto the disc. The file will be called MYPROG. Subsequent execution will start at location 1310.

★ SAVE MYPROG 12FF+100

will load the same area of memory, but this command assumes subsequent execution and reload addresses will be the same as the start address 12FF.

Abbreviation:

★ $. MYPROG 12FF+00

performs the same operation as the last example.

★ SPOOL <fsp>

This command opens a disc file of the name given by <fsp> and any information displayed on the screen is then stored on disc under that filename. The process is cancelled when another ★ SPOOL appears on the screen, either from the keyboard or from a file which is being listed.

Example:

★ SPOOL MYPROG

starts spooling

★ SPOOL

stops spooling

Abbreviation:

★ SP.<fsp>

★ SP.

★ TITLE <disc name>

This command may be used to name or rename a floppy disc.

Example:

★ TITLE UTILITY

★ TITLE "WORK DISC"

Quotation marks are only necessary where there are spaces between letters in the title. Only twelve characters are accepted.

Abbreviation:

★ T!. "WORK DISC"

★ TYPE <fsp>

This performs the same function as ★ LIST, but does not number the lines.

Example:

★ TYPE MYPROG

Abbreviation:

★ TY. MYPROG

As with ★ LIST, this will not display BASIC files as text files. Use ★ SPOOL to convert your BASIC programs text files if you require to use ★ TYPE.

With long listing set page mode by entering <cntrl>N. Reset by entering <cntrl>0.

★ WIPE <AFSP>

This command will delete a group of files from the disc in much the same way as ★ DESTROY. However ★ WIPE prompts you to confirm each file deletion.

Example:

★ WIPE MYPROG #

will delete all files beginning with MYPROG on the current drive and directory, unless they have write protection through locking (see ACCESS) or the disc itself is write-protected.

Abbreviation:

★ W. MYPROG #

RANDOM ACCESS DISC FILES

These files provide one of the powerful features of the BBC computer, and they are one of the major advantages of the disc drive over the tape system.

When you store a long list of data on a tape, for example the names, addresses and telephone numbers of all your friends, there is no way you can jump to one name and telephone number which appears half-way through the list. The whole list must be read from the tape until you reach the name you want. The longer the list, the longer the search is likely to take.

However, with the disc system you can move to precisely the right record and read the relevant data from that record very quickly. Random access files are quick, easy and very useful.

To allow this form of record access, the DFS provides three aids called PTR#, EXT# and EOF#. PTR# points to the next character in a file which is to be read or the next space which is to be used to store a character.

EXT# gives the number of bytes which have been allocated to the file.

EOF# is a flag which becomes TRUE (–1) if the end of the file has been reached, and FALSE (Ø) if it has not. These and other file keywords are summarised below.

The BASIC keywords used to control random access disc files are:

BGET#	OPENIN
BPUT#	OPENOUT
CLOSE#	PRINT#
EOF#	PTR#
EXT#	
INPUT#	

All these keywords are explained in the BBC User Guide.

BGET# <channel number>
BPUT# <channel number>,<number>

BGET# and BPUT# are commands to respectively read and write one byte of a file. The channel to the file must have previously been opened by OPENIN (if reading from the file) or OPENOUT (if writing is required). BGET# requires the channel number as a parameter, BPUT# requires both the channel number and the byte to be stored. The byte may have any value between Ø and 255. If you attempt to write a larger number to the file, only the remainder after the number has been divided by 256 will be stored.

CLOSE# <channel number>

This command will close the specified channel. No more data may be read from or written to the file. If the channel number 0 is used, all channels will be closed.

CLOSE causes all data waiting for transfer to the disc to be sent there. The channel is then free for use with another file. See under OPENIN and OPENOUT.

EOF# <channel number>

When this function is called as shown below, it returns a value of –1 if the file end has been reached, or 0 if it has not. The file must have previously opened with an OPENIN or OPENOUT.

Example:

```
100 X = EOF# 1
```

If the end of file has been reached on channel 1, X will be –1 after this statement.

EXT# <channel number>

This will return the length of the file being accessed through the channel whose number is given as a parameter.

Example:

```
100 X = EXT# 1
```

After this statement has been executed, the value of X will be the allocated length of the file in bytes. The file must have been opened by OPENIN or OPENOUT.

INPUT# <channel number>, <data1>, (<data2>), (<data3>), etc
PRINT# <channel number>, <data1>, (<data2>), etc

INPUT# reads data from an opened file. The data items given in the parameter list of the command may be numerical variables or strings. PRINT# is the corresponding write statement.

OPENIN ("MYDATA")
OPENOUT ("MYDATA")

These two commands are similar in that they both open channels of access between the computer and files on the disc. OPENIN will open a channel for reading only, and OPENOUT will open a channel for writing. The command is used in the form of a function, which returns the value of the channel which has been allocated to the communication with that file.

Example:

X = OPENIN ("MYDATA")

After execution of this statement, X will have the value of channel allocated to reading the file MYDATA. If MYDATA does not exist, X will be Ø.
When you use OPENOUT, a slightly more complex procedure is performed. OPENOUT will delete any file of the name given as the parameter of the command. Then it will create a file of that name, and reserve 64 sectors for the file. (if there was a file of that name already, then the new file will be of the same length). If 64 sectors are not available, then an error is produced, giving the disk full message.

Both OPENIN and OPENOUT commands set PTR# to Ø and OPENOUT sets EXT# to Ø too. OPENIN sets EXT# to the length of the file which has been opened.

The first sector (or currently accessed sector) of the file is loaded into a specially allocated section of memory called the 'buffer'. When writing a file, the data is only actually sent to the disc when the buffer is full or when the file is closed.

There are a maximum of five buffers, so only five channels may be open at the same time.

PTR# <channel number>

This function will give you the location in the file of the next data item that will be either read from or written to. The position is given in bytes relative to the start of the file (location 00). The channel must have been opened by an OPENIN or OPENOUT.

It is important when using random access files to note where each record is located so that PTR# can be set to that location and the data accessed. This is not as difficult as it sounds at first, it only means that your records should be a standard length and each item of data should start at the same location within the record. To ensure this, you must know how the DFS stores data on the disc.

1. Real numbers take up 6 bytes. The first byte defines the type of the data, in this case 'real' which is given the code 255 (FF in hex). The next 5 bytes contain the number in exponential format.

2. Integers take only 5 bytes. The first byte is the code for integers, which is 64 (40 in hex).

3. Strings take up the length of the string + 2 bytes. The first bytes is the code for string, which is 0. The second byte is the length of the string. The string is stored in reverse order of characters.

Reserving space for random access files.

If we started with a new disc and opened two files of different names, then the space allocated would be the first 64 sectors (after the catalogue) for the first file and the next 64 for the second. This would be a problem if the first file was required to be longer than 64 sectors.

An easy way round this is to open the first file, write dummy data to it until it has been extended to the length you require, and then open the second file. This will work on a new disc which has no other files on it.

In reversing space on a used disc, you can utilise the command ★ SAVE to create the reserved space on disc.

```
★ SAVE "MYDATA" 0 0800 0
```

will create a file of 128 sectors long called MYDATA, assuming there are 128 sequential sectors free. When you then subsequently OPENOUT ("MYDATA"), this will clear the file MYDATA and recreate it retaining the length of 128 sectors.

NOTES

SUMMARY OF FILE SYSTEM COMMANDS

APPENDIX